AIP Conference Proceedings

A methodology for parallel implementation of the basic operations of digital signal processing

Cite as: AIP Conference Proceedings **2176**, 030007 (2019); https://doi.org/10.1063/1.5135131
Published Online: 19 November 2019

O. V. Klimova

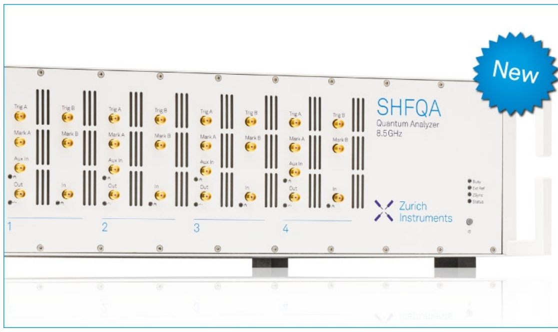
View Online
Export Citation

ARTICLES YOU MAY BE INTERESTED IN

Precision nonlinear low current meter
AIP Conference Proceedings **2176**, 020003 (2019); https://doi.org/10.1063/1.5135115

Numerical study of travelling wave type solutions for the nonlinear heat equation
AIP Conference Proceedings **2176**, 030006 (2019); https://doi.org/10.1063/1.5135130

The boundary value problem of determining hydrogen concentration and the stress state in a titanium shell
AIP Conference Proceedings **2176**, 030005 (2019); https://doi.org/10.1063/1.5135129



Your Qubits. Measured.
Meet the next generation of quantum analyzers
- Readout for up to 64 qubits
- Operation at up to 8.5 GHz, mixer-calibration-free
- Signal optimization with minimal latency

Find out more
Zurich Instruments

© 2019 Author(s).

A Methodology for Parallel Implementation of the Basic Operations of Digital Signal Processing

O. V. Klimova[1, 2]

[1]Institute of Engineering Science, Ural Branch of the Russian Academy of Sciences, 34 Komsomolskaya St., Ekaterinburg, 620049, Russia
[2]B. N. Yeltsin Ural Federal University (UrFU), 19 Mira St., Ekaterinburg, 620002, Russia

Corresponding author: ovk31@mail.ru

Abstract. A methodology for parallel implementation of the basic operations of digital signal processing is considered. The methodology creation is based on the analysis and generalization of the results obtained in the construction of the model description of computation organization. The methodology provides a set of formal transformations that allow you to transform a sequential computing system into a parallel adaptive processing mode. The methodology offers a formal basis for the concurrent exploration of algorithms and architectures, thus creating a basis for improving the efficiency of parallel computing.

INTRODUCTION

Parallel processing has changed the computing environment, which has become two-dimensional, space-time. This resulted in the variety of variants of computation organization and the appearance of the problem of a well-founded choice of a rational variant. The complexity of this problem still hinders the search for its solutions. This complexity is determined by the necessity to change not only the basic components of a computing system (algorithms and architectures), but also relations between them. These relations should be raised to some abstract level, that is, they should be formal, capable of ensuring the implementation of concurrent exploration of algorithms and architectures. At the present state of the art in parallel computing, this research stage is an integral part [1–3] of the process of designing advanced computation systems. Similar to Algorithms + Data Structures = Programs [4], a well-known expression associated with sequential data processing, the authors of [2] gave the expression Algorithm + Architecture = Design, which characterizes the innovative view of parallel computing and demonstrates the inextricable connection between algorithms and architectures. However, a usual algorithmic form was used in that expression. This form was not originally intended to describe the requisite variety. For this, a form is needed that provides the possibility of synthesizing and analyzing variants of computation organization and is characterized by optionality.

Parallel processing gave the right to choose the computation structure at the stage of its description, thereby raising the design process to a higher level of abstraction. It is due to the potential possibility of choosing that we can look for rational variants of implementing computations within the framework of the algorithmic stage of concurrent explorations of algorithms and architectures. However, in order to exercise this right, it must first be described. This can be done by using a description model of computation organization. This form is characterized by optionality, and it can be a basis for the implementation of concurrent explorations of algorithms and architectures. Then the new expression Model + Architecture = Design can be a basis for making a justified choice in the design process, thus generating various parallel implementations of computation. The paper presents a methodology for parallel implementation of the basic operations of digital signal processing (DSP), which creates a common formal basis for making a reasonable choice of variants of parallel computing organization. The methodology was created in the process of analyzing and generalizing the results reported in [1, 5–7], which were obtained when developing a model description of computation organization for a class of DSP operations.

Mechanics, Resource and Diagnostics of Materials and Structures (MRDMS-2019)
AIP Conf. Proc. 2176, 030007-1–030007-4; https://doi.org/10.1063/1.5135131
Published by AIP Publishing. 978-0-7354-1923-0/$30.00

JUSTIFYING THE NEED FOR METHODOLOGY DEVELOPMENT

The development of an independent way of constructing parallel algorithms, initially focused on parallel processing and associated with the study of the intrinsic structure of computational operations and data being processed, led to an evolutionary transition [5] from algorithms to the computation organization model for DSP operations. During the transition, conditions were found under which the expression Algorithm + Data Structure = Model becomes equitable, and new formal tools [6] were developed that allow equivalent transformations to be performed over the original DSP operation expressions. The composition forms obtained as a result of the transformations performed can be considered a parametric expansion of their original forms, which can generate both well-known sequential algorithms and many parallel computational structures. Such composition forms are a tool for describing the structural diversity of algorithms. The construction of these forms means a transition to the model description of the computation organization variants, which allows one to perform their formal synthesis and choice in the process of concurrent explorations of algorithms and architectures. In order to ensure the possibility of implementing this process, it is necessary to develop a system of principles, rules, and methods ensuring the transformations of computing systems of sequential processing into a system of a justified choice of the variant of parallel computation organization. In other words, it is necessary to develop a methodology for parallel implementation of computational operations. A distinctive feature of the required methodology is its directedness from algorithm to architecture (top-down). Then the introduction of such a methodology in the design process will allow us to understand and estimate the expected architecture, as well as to evaluate the entire design space, i.e. all possible design variants.

A METHODOLOGY FOR PARALLEL IMPLEMENTATION OF DSP OPERATIONS

As a result of the analysis of the created model of computation organization for DSP operations, two principles were found that characterize the performed computing system transformations. The essence of one principle is computing environment alteration accompanied by a change in the forms of operation and data representation. The essence of the other one is a change in the algorithm-architecture relationships, which enables algorithmic and architectural parameters to be combined in the model description. The required methodology is characterized by the implementation of these principles, and it can be represented by the following actions:

- using new formal tools allowing you to transfer input data from the time domain to the space-time domain, $t \to (j, t_1)$, and thereby to ensure the possibility of their parallel processing, $x(t) \to x(j, t_1)$;
- using new formal tools specifically designed to represent computational operations with the use of compositional forms and oriented to computations in a space-time environment;
- introducing and using a parameterized coordination-computing environment (CCE) to describe the computation organization;
- mapping the parametrized two-dimensional internal structure of computational operations onto the coordination-computing environment and forming the CCE to represent DSP operations;
- exploring the CCE based on the model description by synthesis and analysis of parallel implementations (of parallel algorithm structures), thus creating a formal basis for a reasonable choice of the parallel implementation variant.

From the above actions, we will form two classes functionally oriented towards transition into the space-time environment and computation organization in this environment, respectively. Then, one class is characterized by the use of new formal tools, and the other class is characterized by the use of coordination-computing environments. To present the methodology, we give a description of these classes.

As a general characteristic of the formal tools used in the methodology, we note that they were developed in the process of studying the influence of the scale of data and operations on the change in their spatial structures. Herewith, the laws of the formation of their internal structure were established, for which purpose components were selected that are invariant to shift and time scaling and allow the original data and operation structures used in sequential processing to be represented by means of compositional forms. As a result, the following formal tools were developed:

- data decomposition methodology oriented to data of any dimension;
- the law describing the internal compositional data structure;

- methods of synthesizing compositional forms for computational operations of the DSP class and operations structurally similar to them.

It is the development of these tools, described in detail in [1, 5–8], that enabled the creation of a methodology for parallel implementation of computations for DSP operations.

Consider the following class of actions based on the obtained possibility of a parameterized description of the computation organization in a space-time environment. These actions are associated with the implementation of the latter of the two above-discussed principles of the methodology in question, which is characterized by a change in the algorithm-architecture relationships. The implementation of these changes became possible due to the obtained possibility of combining algorithmic and architectural parameters in a single model description of the computation organization. Indeed, the transition to compositional forms of operations (CFO) has allowed us to form the following model description of the algorithms: $CFO(A_i^{jp}(t_1), CCE_i(j,p))$. Its components are the algorithms $A_i^{jp}(t_1)$ obtained as a result of decomposition, immersion into the spatial environment, and time compression of the sequential algorithms $A_i(t)$, as well as the coordination-computing environments $CCE_i(j,p)$. In the formation of these components, the following parameters are used: $t = 0, ..., N-1$, where N is the length of the input sequence, $L = N / h_1$, $t_1 = 0, ..., h_1-1, j, p = 0, ..., L-1$.

Figure 1 illustrates the parameterized compositional forms obtained. A distinctive feature of these forms of operations is the introduction and use of coordination-computation environments responsible for coordinating and assembling the results of computations obtained during the implementation of the algorithms $A_i^{jp}(t_1)$.

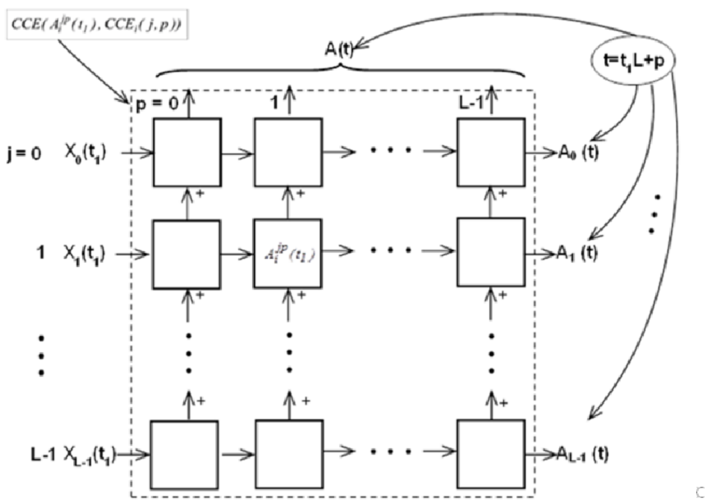


FIGURE 1. Mapping of the internal structure of DSP operations onto the coordination-computing environment

We represent the actions that allow the principle of changing the algorithm-architecture relationships to be implemented. First of all, changing the form of describing computation organization, it is necessary to change their domain of definition. The coordination-computing environment must become such an area for the CFO. Therefore, the first of the above actions is characterized by the introduction and use of such an environment $CCE(j,p,CB(t_1))$. The parameters (j,p) form a two-dimensional spatial structure of this lattice-like environment, in whose nodes the computing blocks $CB(t_1)$ are located, which are responsible for processing the input data $x_j(t_1)$. Having mapped the two-dimensional internal structure of computational operations (CFO) onto the coordination-computing environment $CCE(j,p,CB(t_1))$, we thereby implement the second action of the class under consideration and simultaneously determine the range of the parameters j, $p = 0, ..., L-1$, the type of the computational block $CB(t_1) = A_{mi}^{jp}(t_1)$, and the nature of coordination actions asked by $CCE_i(j,p)$. The result of this mapping is a coordination-computing environment $CCE(A_i^{jp}(t_1), CCE_i(j,p))$ presented in Fig. 1 and describing the variety of possible implementations of parallel computations. It should be noted that such a representation of the coordination-computing environment is the same for the class of DSP operations (convolution, correlation, discrete Fourier transform (DFT)) and operations structurally similar to them. The resulting description represents a homogeneous computing environment characterized by a capacity for scaling, for adjustment by parameters onto different degrees of parallelism, and, consequently, for the synthesis of various variants of computation organization. The development of the description allows us to implement the latter of the above actions characterizing the methodology presented. This action consists in carrying out the explorations of the coordination-computing environment on the basis of the model description by synthesis and analysis of parallel implementations (of parallel algorithm structures) and in creation a formal basis for a justified choice of the parallel implementation variant. The created research ground allows us to implement the above-mentioned principle of changing the algorithm-architecture relationships, which characterizes the methodology under consideration. Indeed, the parametrized form obtained for describing the computation implementation variants assumes its ability to be expanded by introducing into the model the architectural parameters establishing the proposed restrictions on the area used to implement the computations.

CONCLUSION

The methodology presented has been developed based on the analysis of the created model description of parallel algorithm structures for DSP operations. As a result of this analysis, two basic principles have been discovered that ensure the transformation of computing systems of sequential processing into a system of well-founded choice of parallel computation organization variants. The essence of these principles is to change the forms of representation of operations and data and to change the algorithm-architecture relationships. The methodology was formed in the process of creating a rule system aimed at implementing these principles. A distinctive feature of the methodology is its directedness from algorithm to architecture (top-down). The introduction of this methodology into the design process will allow one to evaluate the entire design space and to identify variants that are most interesting for implementation. Moreover, it will be possible to do this at the research stage and at the algorithmic level.

REFERENCES

1. O. V. Klimova, Tomsk State University Journal of Control and Computer Science **2** (23), 112–120 (2013).
2. G. G. Lee, Y. K. Chen, M. Mattavelli, and E. S. Jang, IEEE Trans. Circuits and Systems for Video Technology **19** (11), 1576–1587 (2009).
3. GwoGiun (Chris) Lee, He-Yuan Lin, Chun-Fu Chen, and Tsung-Yuan Huang, IEEE Trans. on Parallel and Distributed Systems **23** (5), 944–957 (2012).
4. N. Wirth, *Algorithms+Data Structures=Programs* (Prentice Hall PTR, Upper Saddle River, USA, 1978).
5. O. V. Klimova, Tomsk State University Journal of Control and Computer Science **1**, 31–38 (2011).
6. Tomsk State University Journal of Control and Computer Science **2**, 11–22 (2016).
7. O. V. Klimova, Izvestiya Vysshikh Uchebnykh Zavedeniy. Fizika **8** (2), 49–52 (2016).
8. O. V. Klimova, Journal of Computer and Systems Sciences International **38** (3), 402–408 (1999).